Міністерство освіти і науки України
Національний університет «Львівська політехніка»

Звіт
з лабораторної роботи №3
з дисципліни: “Кросплатформні засоби програмування”
на тему: “Спадкування та інтерфейси”

Варіант 8

Виконав:
Киянець А.М.
Студент групи КІ- 306
Прийняв:
Іванов Ю.С.

Львів 2023

Мета

Ознайомитися зі спадкуванням та інтерфейсами у мові Java.

Індивідуальне завдання

ЗАВДАННЯ

1. Написати та налагодити програму на мові Java, що розширює клас, що реалізований у лабораторній роботі №2, для реалізації предметної області заданої варіантом. Суперклас, що реалізований у лабораторній роботі №2, зробити абстрактним. Розроблений підклас має забезпечувати механізми свого коректного функціонування та реалізовувати мінімум один інтерфейс. Програма має розміщуватися в пакеті `Група.Прізвище.Lab3` та володіти коментарями, які дозволять автоматично згенерувати документацію до розробленого пакету.
2. Автоматично згенерувати документацію до розробленого пакету.
3. Завантажити код на GitHub згідно методичних вказівок по роботі з GitHub.
4. Скласти звіт про виконану роботу з приведенням тексту програми, результату її виконання та фрагменту згенерованої документації та завантажити його у ВНС.
5. Дати відповідь на контрольні запитання.

Написати та налагодити програму на мові Java, що розширює клас, реалізований у лабораторній роботі №3, для реалізації предметної області заданої варіантом: Цифрова відеокамера

Хід роботи

1. Запустив середовище та написав програму згідно індивідуального завдання

Camera.java

```
package KI306.Kyianets.Lab3;

import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;

public abstract class Camera {
    private String model;
    private Lens lens;
    private Sensor sensor;

    // Абстрактний конструктор
    public Camera(String model, Lens lens, Sensor sensor) {
        this.model = model;
        this.lens = lens;
        this.sensor = sensor;
    }

    // Абстрактний метод для отримання інформації про камеру
    public abstract String getInfo();

    // Геттер для отримання моделі камери
    public String getModel() {
        return model;
    }

    // Геттер для отримання об'єктиву
    public Lens getLens() {
        return lens;
```

```
    }

    // Геттер для отримання сенсора
    public Sensor getSensor() {
        return sensor;
    }

    // Метод для запису повідомлень в лог-файл
    public void writeToLog(String message) {
        try (PrintWriter writer = new PrintWriter(new FileWriter("log.txt",
true))) {
            writer.println(message);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    // Абстрактний метод для коректного завершення роботи
    public abstract void close();
}
```

Main.java

```
package KI306.Kyianets.Lab3;

public class Main {
    public static void main(String[] args) {
        // Створюємо об'єкт WaterproofCamera
        Lens lens = new Lens("50mm", "f/1.8");
        Sensor sensor = new Sensor("Full Frame", "20MP");
        WaterproofCamera waterproofCamera = new WaterproofCamera("Canon EOS",
lens, sensor, false);

        // Тестуємо методи
        waterproofCamera.writeToLog("WaterproofCamera initialized.");
        System.out.println(waterproofCamera.getInfo()); // Метод getInfo з
абстрактного класу
        waterproofCamera.makeWaterProof(); // Метод з інтерфейсу WaterProof
        System.out.println(waterproofCamera.getInfo());
        waterproofCamera.makeBullerProof();
        System.out.println(waterproofCamera.getInfo());
        waterproofCamera.close(); // Метод з абстрактного класу
    }
}
```

Lens.Java

```
package KI306.Kyianets.Lab3;

public class Lens {
    private String focalLength;
    private String aperture;

    public Lens(String focalLength, String aperture) {
        this.focalLength = focalLength;
        this.aperture = aperture;
    }

    public String getFocalLength() {
        return focalLength;
    }
```

```
    // Новий метод для отримання значення апертури
    public String getAperture() {
        return aperture;
    }

    // Додайте інші методи та конструктори за необхідності.
}
```

Sensor.java

```
package KI306.Kyianets.Lab3;

public class Sensor {
    private String type;
    private String resolution;

    public Sensor() {
        // Конструктор без параметрів
    }

    public Sensor(String type, String resolution) {
        this.type = type;
        this.resolution = resolution;
    }

    public String getType() {
        return type;
    }

    // Новий метод для отримання значення роздільної здатності
    public String getResolution() {
        return resolution;
    }

    // Додайте інші методи та конструктори за необхідності.
}
```

WaterproofCamera.java

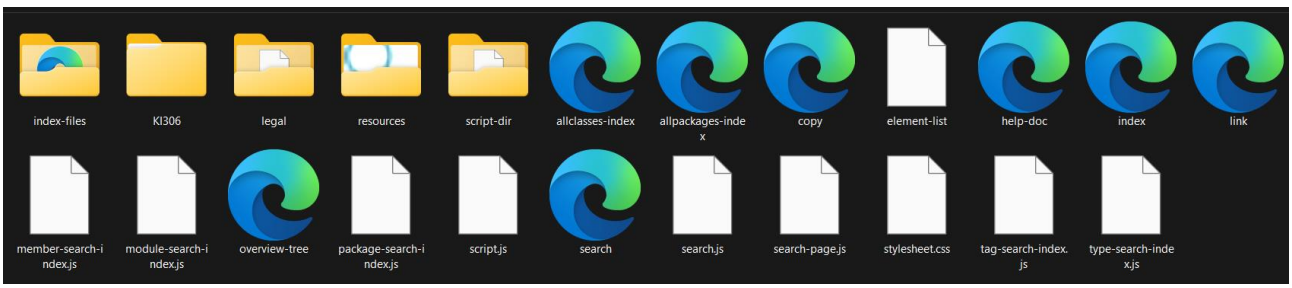
```
package KI306.Kyianets.Lab3;

// Інтерфейс
interface WaterProof {
    void makeWaterProof();
}

interface BulletProof {
    void makeBullerProof();
}
// Підклас, що реалізує суперклас та інтерфейс
public class WaterproofCamera extends Camera implements WaterProof ,BulletProof{
    private boolean isWaterproof;

    private boolean isBulletproof;
    public WaterproofCamera(String model, Lens lens, Sensor sensor, boolean
isWaterproof) {
        super(model, lens, sensor);
        this.isWaterproof = isWaterproof;
    }

    // Реалізація абстрактного методу
    @Override
```

```
    public String getInfo() {
        String waterproofInfo = isWaterproof ? "Waterproof" : "Not Waterproof";
        return getModel() + " | " + getLensInfo() + " | " + getSensorInfo() + "
| " + waterproofInfo+ " | " +"Bulletproof="+isBulletproof;
    }

    // Реалізація інтерфейсу
    @Override
    public void makeWaterProof() {
        isWaterproof = true;
        writeToLog("Camera made waterproof.");
    }

    // Реалізація абстрактного методу
    @Override
    public void close() {
        // Додайте код для коректного завершення роботи
        writeToLog("Camera closed.");
    }

    // Додатковий метод для отримання інформації про об'єктив
    private String getLensInfo() {
        return "Lens: " + getLens().getFocalLength() + " | " +
getLens().getAperture();
    }

    // Додатковий метод для отримання інформації про сенсор
    private String getSensorInfo() {
        return "Sensor: " + getSensor().getType() + " | " +
getSensor().getResolution();
    }

    @Override
    public void makeBullerProof() {
        isBulletproof = true;
        writeToLog("Camera made bulletproof.");
    }
}
```

Запуск програми після захисту роботи

```
"D:\Software\Java\jdk 21\bin\java.exe" "-javaagent:D:\Software\IntelliJ IDEA 2023.2.5\lib\idea_rt.
Canon EOS | Lens: 50mm | f/1.8 | Sensor: Full Frame | 20MP | Not Waterproof | Bulletproof=false
Canon EOS | Lens: 50mm | f/1.8 | Sensor: Full Frame | 20MP | Waterproof | Bulletproof=false
Canon EOS | Lens: 50mm | f/1.8 | Sensor: Full Frame | 20MP | Waterproof | Bulletproof=true
```

Сформована Java документація



Висновок

На даній лабораторній роботі ознайомився зі спадкуванням та інтерфейсами у мові Java.